GSoC 2017 Project Proposal

Name : Yoginth Saravanan
Country : India
College : Dr.Mahalingam College of Engineering and Technology, Pollachi.
Degree : Bachelor of Engineering in Computer Science
Email : yoginth@zoho.com
Website : https://yoginth.zoho.com
Blog : https://yoginth.tumblr.com
Github : https://github.com/yoginth
Archive Profile : https://archive.org/details/@yoginth
Phone: +91-7010851615

Title : Add features to the existing "Wayback Machine" Chrome extension.

Introduction:

In this Year of Google Summer of Code I Would like to work on those areas given below in "Wayback Machine" which archives entire internet

- Integration with the Wayback Machine's Site Search
- Support for social sharing, including Twitter and Facebook
- Display Tweets about a URL for a given date range
- Provide user with a one-click Summary view of a given site (Alexa Rank)
- Automatically archive URLs that are not in the Wayback Machine

Background:

There are only a limited number of options that have been implemented in Wayback Machine. It's still in the growing phase. Addition of the operations given below to Wayback Machine will help the users to view the history of the web pages or even their blog and see the webpage statistics like Alexa Ranking etc.

Idea:

i) Integration with the Wayback Machine's Site Search

On the World Wide Web, a query string is the part of a uniform resource locator (URL) containing data that does not fit conveniently into a hierarchical path structure.

The query string commonly includes fields added to a base URL by a Web browser or other client application, for example as part of an HTML form.

A typical URL containing a query string is as follows:

http://web.archive.org/web/*/{user_query}

{user_query} = User Entered Query String it may be an URL or an Keyword. With Simple HTML **GET** and **POST** method is used to Pass the URL or Keyword to the server via URL. And i will make more Secured by Encrypting the String using Some **cipher** method.

ii) Support for social sharing, including Twitter and Facebook

With the help of Twitter and Facebook Social Sharing API we can Include the social sharing the archived website even to Private Message in Facebook. Thanks to Twitter API which is to be included in Wayback Machine Chrome Extension. The simple Script tag is given below.

```
<a href="https://twitter.com/share" class="twitter-share-button"
data-url="http://web.archive.org/web/20070309185314/http://www.google.com/"
data-size="large">Tweet</a>
<script>!function(d,s,id){var
js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+'://platform.twitter.com/widgets.js';fjs.parentNode.insertBefore(js,fjs);}}(document, 'script', 'twitter-wjs');</script>
```

iii) Display Tweets about a URL for a given date range

Again a Big Thanks to **Twitter** to provide search **API** for tweets. And we should note that Twitter's search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface. In API v1.1, the response format of the Search API has been improved to return Tweet objects more similar to the objects you'll find across the REST API and platform. However, perspectival attributes (fields that pertain to the perspective of the authenticating user) are not currently supported on this endpoint.

Brief Note : https://dev.twitter.com/rest/reference/get/search/tweets

iv) Provide user with a one-click Summary view of a given site (Alexa Rank)

We can Implement the Alexa's Rank of a Particular site by the same method which is used in Search but here we implement it in different way to implement it in the Extension, Here we will pass the URL of a webpage via Alexa's Traffic URL and the informations are Gathered fro Alexa's Server and Display it in The Chrome Extension,

The url will be like this

http://traffic.alexa.com/graph?&u=twitter.com

v) Automatically archive URLs that are not in the Wayback Machine

At first we need the check the webpage that is available in Wayback Machine Server or not, so thanks to wayback's own api to check it with the help of this url

http://archive.org/wayback/available?url=google.com

If it is available it will return some values like

{"archived_snapshots":{"closest":{"available":true,"url":"http://web.archive.org/web/20170331143429/http://www.google.com/","timestamp":"20170331143429","status":"200"}}}

If it is not available it will return nothing

http://archive.org/wayback/available?url=somedifferentsites.com

Like this

{"archived_snapshots":{}}

So with the Help of this we can Implement the auto archiving easily by running the URL automatically.

By Running the given url

https://web.archive.org/save/http://somedifferentsites.com

Timeline:

Before April 23: Self study

- Familiarize myself completely with Wayback Machine's functions and architecture.
- Understand the algorithms that are already implemented in Wayback Machine's Chrome Extension.
- Some of the algorithms have been implemented by me, so will finish off studying the remaining ones.
- Get through the tutorials quickly.

April 23 - May 21: Internet Archive Community Bonding

- To do self coding with Wayback Machine to improve my further understanding and ease of use with the software.
- Experimentation and testing with Wayback Machine's development Extension.
- During this period, I will remain active on IRC and Mailing List to discuss and finalize on how to proceed with the project. Create a github repo where all the code will be pushed once the testing of implemented algorithms is done.(https://github.com/yoginth/waybackmachine)
- Documentation can be done using Read The Docs.

May 21 - May 28:

- Testing them on different platforms and in different versions of chrome.

May 29 - June 6:

- Optimizing
- Testing, debugging.

June 7 -June 14:

- Implement Twitter and Facebook APIs.
- Compare and test them.

June 15 -June 22:

- Implement Wayback machine APIs.
- Compare, test and debug.

June 23 -June 30:

- Implement Image Transformation Methods in Alexa Traffic Graph.
- Making Great GUI with best UI and UX.
- Test the working of these operations.

July 1 -July 8:

- Documentation of what i have implemented.
- Debugging.

July 9 -July 13: (Mid Term Evaluation)

- Testing the overall working of each and every module/function.

July 14 -July 21:

- Implement Bootstrap
- Testing on various dataset using Selenium.

July 22 -July 29:

- Implement Node.js

- Testing on Wear and Tear.

July 30 -Aug 6:

- Develop GUI.
- Testing, Debugging

Aug 7 -Aug 13:

- Documentation
- Testing
- Results and comparisons of different algorithms on different datasets.
- Cleaning and Optimizing the unused and died codes.

Aug 13 -Aug20: (Pencils down date)

- Take a week to scrub code, write a few test cases and test them, improve the current documentation.

Aug21 –Aug24:

- Buffer period. (Add Tutorials on implemented functionalities)
- According to the time line, algorithms will be implemented before the beginning of the final evaluation.

Algorithms to be implemented after GSoC or in case the above algorithms are finished before time:

- Say Goodbye to **404 Error**
- Implementing WHOIS Information
- Feedback Section

General computing experience:

Operating System : GNU/LINUX, Windows
Programming Languages : C, C++
Scripting Languages : Python, PHP
Internet technologies : HTML, CSS, Javascript
Server Side Scripting : Mod_python, Mod Apache

Database management System : MySQL
Graphics and UI Development : OpenGl, Bootstrap, AngularJS
Programming Environments : VIM, Emacs, Atom, Cloud9, TurboC3
Other Tools : Github, Gulp, Trello, Heroku